

WHAT IS CLAIMED IS:

1. An intermediate object linking method of linking
apluralityofintermediateobjectstoformanexecutableobject,
5 comprising:

an intermediate object linking order forming step which
decides linking orders of the plurality of intermediate objects;

a linking processing step which executes linking
processes of the plurality of intermediate objects based on
10 the linking orders decided by the intermediate object linking
order forming step to get an executable object;

a comparing step which compares program size of the
executable objects obtained by the linking processing step with
the program size of a executable objects stored in a storing
15 section every time when the linking order is changed;

a storing step for storing the program size of the
executable objects and the linking order obtained by the linking
processing step in the storing section to update when the program
size of the executable objects obtained by the linking processing
20 step is smaller than the program size of the executable objects
stored in the storing section at the comparing step; and

a repeating step for changing the linking orders by the
intermediate object linking order forming step and executing
repeatedly the linking processing step, the comparing step,
25 and the storing step.

2. An intermediate object linking method according to claim 1, wherein the intermediate object linking order forming step decides the linking orders by a permutation algorithm.

5

3. An intermediate object linking method according to claim 1, wherein the intermediate object linking order forming step decides the linking orders by a genetic algorithm.

10 4. An intermediate object linking unit for linking a plurality of intermediate objects to form an executable object, comprising:

15 an intermediate object linking order forming section which decides linking orders of the plurality of intermediate objects;

a linker starting section which executes linking processes of the plurality of intermediate objects based on the linking orders decided by the intermediate object linking order forming section to form an executable object;

20 a comparing section which compares program size of the executable objects every time when the linking order is changed;

a storing section which stores the program size of the executable objects and the linking order; and

25 a repeating section which changes the linking orders by the intermediate object linking order forming section and

operating repeatedly the linker starting section, the comparing section, and the storing section,

wherein the comparing section compares the program size of the executable objects formed by the linker starting section
5 with the program size of the executable objects stored in the storing section,

wherein the storing section stores the program size of the executable objects and the inking order formed by the linker starting section when the program size of the executable objects
10 formed by the linker starting section is smaller than the program size of the executable objects stored in the storing section at the comparison by the comparing section.

5. An intermediate object linking unit according to
15 claim 4, wherein the intermediate object linking order forming section decides the linking orders by a permutation algorithm.

6. An intermediate object linking unit according to
20 claim 4, wherein the intermediate object linking order forming section decides the linking orders by a genetic algorithm.

7. A linker unit for linking a plurality of intermediate objects to form an executable object, comprising:
an intermediate object linking order forming section
25 which decides linking orders of the plurality of intermediate

objects;

a linking processing section which executes linking processes of the plurality of intermediate objects based on the linking orders decided by the intermediate object linking order forming section to form an executable object;

a comparing section which compares program size of the executable objects every time when the linking order is changed;

a storing section which stores the program size of the executable objects and the linking order; and

a repeating section which changes the linking orders by the intermediate object linking order forming section and operating repeatedly the linker starting section, the comparing section, and the storing section,

wherein the comparing section compares the program size of the executable objects formed by the linker starting section with the program size of the executable objects stored in the storing section,

wherein the storing section stores the program size of the executable objects and the linking order formed by the linker starting section when the program size of the executable objects formed by the linker starting section is smaller than the program size of the executable objects stored in the storing section at the comparison by the comparing section.

8. A linker unit according to claim 7, wherein the

intermediate object linking order forming section decides the linking orders by a permutation algorithm.

9. A linker unit according to claim 7, wherein the
5 intermediate object linking order forming section decides the linking orders by a genetic algorithm.

10. A compiler driving unit for translating a source
program by starting a compiler, an assembler, a linker, etc.
10 to form an executable object, comprising:

an intermediate object linking order forming section
which decides linking orders of the plurality of intermediate
objects;

a linker starting section which executes linking
15 processes of the plurality of intermediate objects based on the linking orders decided by the intermediate object linking order forming section to form an executable object;

a comparing section which compares program size of the
executable objects every time when the linking order is changed;

20 a storing section which stores the program size of the executable objects and the linking order; and

a repeating section which changes the linking orders by
the intermediate object linking order forming section and
operating repeatedly the linker starting section, the comparing
25 section, and the storing section,

wherein the comparing section compares the program size of the executable objects formed by the linker starting section with the program size of the executable objects stored in the storing section,

5 wherein the storing section stores the program size of the executable objects and the inking order formed by the linker starting section when the program size of the executable objects formed by the linker starting section is smaller than the program size of the executable objects stored in the storing section
10 at the comparison by the comparing section.

11. A compiler driving unit according to claim 10,
wherein the intermediate object linking order forming section decides the linking orders by a permutation algorithm.

12 A compiler driving unit according to claim 10,
wherein the intermediate object linking order forming section decides the linking orders by a genetic algorithm.

20 13. A recording medium for recording a program for linking a plurality of intermediate objects to form an executable object, wherein the program for causing a computer to execute a method comprising:

an intermediate object linking order forming step which
25 decides linking orders of the plurality of intermediate objects;

a linking processing step which executes linking processes of the plurality of intermediate objects based on the linking orders decided by the intermediate object linking order forming step to get an executable object;

5 a comparing step which compares program size of the executable objects obtained by the linking processing step with the program size of a executable objects stored in a storing section every time when the linking order is changed;

10 a storing step for storing the program size of the executable objects and the linking order obtained by the linking processing step in the storing section to update when the program size of the executable objects obtained by the linking processing step is smaller than the program size of the executable objects stored in the storing section at the comparing step; and

15 a repeating step for changing the linking orders by the intermediate object linking order forming step and executing repeatedly the linking processing step, the comparing step, and the storing step.

20 14. A recording medium according to claim 13, wherein the intermediate object linking order forming step decides the linking orders by a permutation algorithm.

25 15. A recording medium according to claim 13, wherein the intermediate object linking order forming step decides the

linking orders by a genetic algorithm.